**ExecuteFromArrays.ChildrenRonde Pseudocode:**

int ChildrenRonde (int[] C, integer steps)

if steps <= 0 OR C.length <= 0

print(“There’s nothing here!”)

exit

int cout = 0

for loop i = 1; C.length + 1 times; i++

C[cout] = i

cout = cout + 1

int counter = 0

for loop int i = 0; C.length – 1 times; i++ {

for loop int j = 1; steps times; j++ {

if counter >= C.length

counter = counter - C.length

while C[counter] == 0 {

counter++

if counter >= C.length

counter = counter – C.length

}

counter++

if counter >= C.length

counter = counter – C.length

}

while C[counter] == 0 {

counter++

if counter >= C.length

counter = counter - C.length

}

C[counter] = 0

counter = counter + 1

}

int winner = 0

for loop int i = 0; C.length times; i++) {

if C[i] > 0 {

winner = C[i]

}

}

print("Child #")

return winner

}

**Clist.ChildrenRonde Pseudocode:**

void ChildrenRonde(integer steps) {

Node iter = start

if start == null {

print("List is empty.")

exit

}

if size == 1 {

print("There is only one element remaining.")

iter.print

exit

}

while size != 1 {

for loop int i = 1; steps times; i++ {

iter.next

}

removeNode.iter

iter.next

}

List.print

}

**Test Methods:**

ExecuteWithClist Tests:

1. **Baseline Test (Size 8, Steps 4):** This is a baseline test to check if the functionality of program involving circular linked lists actually works. I used a size of 8 and 4 steps (similar in the example in the lab description).

**EXPECTED:**

Child #6: child

**OUTCOME:**

Child #6: child

2. **Single child test (Size 1, Step 3):** This is more of an edge test to see if the the program can recognize if the size is only one, and that if it can return one only properly.

**EXPECTED:**

Child #1: child

[**OUTCOME:**](https://www.google.com/search?client=firefox-b-1-d&q=dick+cheney&tbm=nws&source=univ&tbo=u&sa=X&ved=2ahUKEwjkufH2z7fhAhUJO60KHQpcDtsQt8YBKAF6BAgAEC4)

Child #1: child

ExecuteWithArrays Tests:

1. **Baseline Test (Size 7, Steps 3):** This is a baseline test to check if the program that handles arrays will run as intended. So the size and steps are normal values.

**EXPECTED:**

Child #4

**OUTCOME:**

Child #4

2. **Zero children test (Size 0, Step 3):** This is a edge test where the size is zero, so this test is to see if the program can properly handle this improper input, and give a proper message to the user.

**EXPECTED:**

**“**The steps are zero or negative, or size is zero (which means empty array). Exiting.”

**OUTCOME:**

“The steps are zero or negative, or size is zero (which means empty array). Exiting.”

**Historical Background:**

This game is based off the Josephus problem where every nth person will be eliminated until there is one person left. The story behind this game/problem is that of a group of men hid in a cave during a revolt against Rome. Fearing the Romans will eventually find them, they all decided to kill themselves by killing every seventh man in a circle they have formed. Josephus, being a mathematician, figured out that he just needed position himself in the right spot so that he’ll be the last man standing. Needless to say, Josephus manage to survive the ordeal.